Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Beyond Uniform: To Boldly Abstract What Has Not Been Abstracted Before

Matthias Knorr[1] , **Zeynep G. Saribatur**[2] , **Ricardo Gonçalves**[1]
[1]NOVA LINCS, NOVA University Lisbon
[2]Institute of Logic and Computation, TU Wien
{mkn, rjrg}@fct.unl.pt, zeynep.saribatur@tuwien.ac.at

Abstract

The ability to abstract is important in AI systems and in problem solving, as generalizing over irrelevant details facilitates finding solutions. In the context of Answer Set Programming (ASP), abstraction has been recently investigated with a focus on the omission of unnecessary details, which is related to forgetting, as well as on clustering vocabulary of similar concepts into a common abstract representation. In the latter case, a characterization has been provided that identifies when such abstraction is possible without affecting the answer sets under the addition of any set of facts, in the spirit of uniform equivalence and aligned with the ASP methodology where a general problem encoding is used with varying instances. However, when this characterization fails, no abstraction is possible, and even if it succeeds, computing an abstracted program syntactically is only possible for a limited subclass of programs. In this paper, we consider that not all kinds of facts are required to be added in general, and investigate under which conditions such abstraction is indeed always possible as well as when and how to compute abstracted programs, generalizing at the same time (compared to related work) to a larger class of programs with wider applicability.

1 Introduction

Human reasoning relies on abstraction and generalization to support flexible decision-making under changing conditions and to simplify complex problem-solving tasks by ignoring irrelevant details. Similar goals arise in AI, where systems must construct representations that allow solving multiple related problem instances and also generalize beyond observed instances. In Symbolic AI, these challenges have mainly been addressed along two complementary directions: learning programs from examples, as pursued in logic-based machine learning approaches such as inductive logic programming (Cropper et al. 2022), whose tools continue to be enhanced and refined (Cerna and Cropper 2024); and abstraction-based approaches, for example in planning, where abstract representations are constructed to enable finding generalized plans (Illanes and McIlraith 2019) or learning generalized policies (Bonet, Francès, and Geffner 2019). However, guarantees on the soundness and completeness of the resulting models are often lacking in learning-based settings, due to potential overfitting or limitations in background knowledge. Although formal guarantees for generalized planning have been investigated (Bonet et al. 2019; Cui, Liu, and Luo 2021; Giacomo, Lespérance, and Mancanelli 2025), it remains unclear how these theoretical results can be transferred to other domains.

In Knowledge Representation and Reasoning, removal of (unnecessary) details in logical representations has been approached with a number of different ideas. Forgetting (Eiter and Kern-Isberner 2018) focuses on finding a representation over a reduced language, thereby omitting information no longer deemed relevant, and has been studied together with closely related notions, including variable elimination (Lang, Liberatore, and Marquis 2003), uniform interpolation (Visser 1996), or ignorance (Baral and Zhang 2005). Another line of research is on representations over a higher-level language that preserve certain properties (Giunchiglia and Walsh 1992; Nayak and Levy 1995; Lomuscio and Michaliszyn 2014; Banihashemi, De Giacomo, and Lespérance 2017). Although there are well-established theoretical approaches to understand the properties needed from abstracted programs, a domain-independent view on how these techniques benefit generalization has only recently gained traction.

For the framework of Answer Set Programming (ASP), this intuition has recently been formalized through the clustering of similar concepts into unified abstract representations (Saribatur et al. 2024). Such abstractions are typically evaluated through the lens of uniform equivalence (Eiter and Fink 2003), which ensures that an encoding remains faithful to its original version regardless of the specific set of external facts, i.e., the instance data, added to the program. This perspective aligns with standard ASP methodology, where a general problem encoding is designed to remain robust across varying input instances. While uniform abstraction can be viewed as an initial step towards a theoretical foundation for generalized reasoning, in the spirit of generalized planning, it often proves to be overly restrictive. Namely, the notion requires an abstraction to hold under the addition of any possible set of facts over the entire language.

In practical scenarios, however, many predicates in a program are auxiliary or internal, and a developer can usually guarantee that only a specific subset of the vocabulary will be provided as input. This mirrors the motivation behind relativized notions of strong/uniform equivalence (Woltran 2004; Eiter, Fink, and Woltran 2007), which recognize that equivalence only needs to hold under contexts relevant to the

program's actual use. This also aligns well with the concept of modularity in ASP (Janhunen et al. 2009), where ASP programs with well-defined input-output interfaces are employed to restrict the language that is used externally and in between them, and where forgetting has been studied as well (**(UP)**-forgetting (Gonçalves et al. 2019)). In either case, abstraction would be desirable, but is prevented by the excessively strong conditions on its applicability.

In this paper, we overcome these limitations by relaxing the notion of uniform abstractions admitting the addition of facts over a specific set of atoms.

Our contributions are summarized as follows.

- We introduce uniform B-m-abstractions to logic programs for clustering atoms in the vocabulary with a mapping m while satisfying relations resembling relativized uniform equivalence w.r.t. a set B of atoms to be added.
- We provide necessary and sufficient conditions for testing whether a program can have relativized uniform abstractions, while introducing a model-based representation to capture these. We also identify conditions for which such abstraction always exists.
- We employ extended logic programs, i.e., disjunctive programs with double negation in the body, which also generalizes previous work in that aspect and broadens practical applicability. In addition, we show for subclasses of programs when applying abstractions is closed.
- We define a syntactic operator for computing abstractions that preserves the original rules as much as possible, but with a considerably larger applicability than a previous operator (Saribatur et al. 2024), and we identify concisely the conditions under which it can be applied.
- We show how the existing notions of uniform simplification and **(UP)**-forgetting can be seen as special cases, by clustering the atoms into $\top$. We show under which conditions these special cases can be applied.
- We provide complexity results for the problems of deciding relativized uniform abstractability and equivalence testing that are often aligned with previous results on abstractions, but affected by the increasing complexity of checking relativized equivalence.

2 Extended Logic Programs

We recall necessary notions and notation on logic programs including the different semantics we employ in this paper. Here, we restrict our considerations to propositional programs over a set of propositional atoms $\mathcal{U}$, while programs with variables can be reduced to their ground versions, corresponding to propositional programs, as usual.

We consider rules r of the form

$$A_1 \vee \cdots \vee A_l \leftarrow A_{l+1}, \ldots, A_m, not\ A_{m+1}, \ldots, not\ A_n, \\ not\ not\ A_{n+1}, \ldots not\ not\ A_o$$

where $0 \leq l \leq m \leq n \leq o$, $A_i \in \mathcal{U}$, for $1 \leq i \leq o$, and not is default negation. We also write r as $H(r) \leftarrow B(r)$ or $H(r) \leftarrow B^+(r), not\ B^-(r), not\ not\ B^{--}(r)$. We call $H(r) = \{A_1, \ldots, A_l\}$ the *head* of r, $B^+(r) = \{A_{l+1}, \ldots, A_m\}$ the *positive body* of r, $B^-(r) = \{A_{m+1}, \ldots, A_n\}$ the *negative body* of r, and $B^{--}(r) = \{A_{n+1}, \ldots, A_o\}$ the *double-negated body* of r. If $H(r) = \emptyset$, occasionally written as $\bot$, then r is a *constraint*; and if $B(r) = \emptyset$ and $l = 1$, then r is a *fact*. A rule r is *disjunctive* if $B^{--}(r) = \emptyset$, *normal* if additionally $l \leq 1$, and *positive* if also $B^-(r) = \emptyset$. A rule is *Horn*, if it is normal and positive.

An *extended (logic) program* (ELP) is a finite set of rules, a *disjunctive program* (DLP) is a finite set of disjunctive rules, and a *normal program* (NLP) is a finite set of normal rules. Unless stated otherwise, the term *program* refers to an extended logic program.

The *(labeled) dependency digraph* $\mathcal{D}_P$ is the digraph defined on the set $\bigcup_{r \in P}(H(r) \cup B^+(r) \cup B^-(r) \cup B^{--}(r))$ of atoms, where for every rule $r \in P$, two atoms $b \in B^+(r) \cup B^-(r) \cup B^{--}(r)$ and $a \in H(r)$ are joined by an edge (b, a) labeled by "$-$" if $b \in B^-(r)$ and "$+$" if $b \in B^+(r) \cup B^{--}(r)$. A *cycle in* $\mathcal{D}_P$ is a sequence $v_1, v_2, \ldots v_n$ of vertices s.t. there is an edge from v_i to v_{i+1}, $1 \leq i \leq n$, and $v_1 = v_n$. An *odd cycle* of P is a cycle in $\mathcal{D}_P$ that contains an odd number of negative labeled edges.

An interpretation I is a set of atoms from $\mathcal{U}$ that are true. As usual, I is a model of a program P, denoted by $I \models P$, if it maps all rules of P to true. The semantics of answer set programs are commonly defined via program *reducts*. The following reduct was defined by Eiter and Wang (Eiter and Wang 2008) and extends the original *GL-reduct* (Gelfond and Lifschitz 1988) to support double negation. The *reduct* of a program P w.r.t. an interpretation I is given by $P^I = \{H(r) \leftarrow B^+(r) \mid r \in P, B^-(r) \cap I = \emptyset, B^{--}(r) \subseteq I\}$, and I is an *answer set* of P if it is a minimal model of P^I. We denote the set of all answer sets of P by $AS(P)$. Two programs P_1, P_2 are *equivalent* if $AS(P_1) = AS(P_2)$, *strongly equivalent (SE)*, denoted by $P_1 \equiv P_2$, if $AS(P_1 \cup R) = AS(P_2 \cup R)$ for every program R over $\mathcal{U}$, and *uniformly equivalent (UE)*, denoted by $P_1 \equiv_u P_2$, when R is restricted to set of facts over $\mathcal{U}$. Relativized notions of strong/uniform equivalence consider a relaxation on the vocabulary of the added programs. Formally, two programs P_1 and P_2 are *strongly equivalent relative to* B, denoted $P_1 \equiv^B P_2$ iff $AS(P_1 \cup R) = AS(P_2 \cup R)$ for all programs R over $B \subseteq \mathcal{U}$, and *uniformly equivalent relative to* B, denoted $P_1 \equiv_u^B P_2$, when R is restricted to set of facts over B. These naturally cover strong and uniform equivalence for $B = \mathcal{U}$.

These equivalence notions have also been captured in model notions founded in the logic of Here-and-There (Pearce 1996), and we recall these next.

An *SE-interpretation* is a pair $\langle X, Y\rangle$ such that $X \subseteq Y \subseteq \mathcal{U}$; it is *total* if $X = Y$ and *non-total* otherwise. An SE-interpretation $\langle X, Y\rangle$ is an *SE-model* of a program P if $Y \models P$ and $X \models P^Y$, and we denote by $SE(P)$ the set of all SE-models of P. A set Y of atoms is also an answer set of P if $\langle Y, Y\rangle \in SE(P)$ and no non-total $\langle X, Y\rangle \in SE(P)$ exists, and two programs P_1 and P_2 are strongly equivalent iff $SE(P_1) = SE(P_2)$ (Turner 2001).

While SE-models capture strong equivalence, relativized SE-models admit considering only a subset B of $\mathcal{U}$.

Definition 1 ((Eiter, Fink, and Woltran 2007)). *A pair of interpretations $\langle X, Y\rangle$ is a (relativized) B-SE-interpretation iff either $X = Y$ or $X \subset (Y \cap B)$. The former are called total and the latter non-total B-SE-interpretations.*

Moreover, a B-SE-interpretation $\langle X, Y\rangle$ is a (relativized) B-SE-model of a program P iff:

(i) $Y \models P$;

(ii) for all $Y' \subset Y$ with $(Y' \cap B) = (Y \cap B)$, $Y' \not\models P^Y$; and

(iii) $X \subset Y$ implies existence of $X' \subseteq Y$ with $X' \cap B = X$, such that $X' \models P^Y$ holds.

The set of B-SE-models of P is given by $SE^B(P)$.

Two DLPs P_1, P_2 are strongly equivalent relative to B iff $SE^B(P_1) = SE^B(P_2)$.

The following Lemma generalizes results on the relation between B-SE-models and SE-models [Lemma 4, (Eiter, Fink, and Woltran 2007)] from DLPs to ELPs.[1]

Lemma 1. *Let P be a program and A be a set of atoms.*

(i) If $\langle Y, Y\rangle \in SE^B(P)$, then $\langle Y, Y\rangle \in SE(P)$.

(ii) If $\langle X, Y\rangle \in SE^B(P)$ with $X \subset Y$, then $\langle X', Y\rangle \in SE(P)$, for some $X' \subseteq Y$ with $(X' \cap B) = X$.

The converse of (i) does not hold in general, as (ii) of Def. 1 imposes an additional condition on total B-SE models. Also, the non-total B-SE-models of a program P are in general indeed not a subset of its non-total SE-models:

Example 1. *Consider program P and $B = \{a, b\}$:*

$$c \leftarrow a \qquad a \leftarrow c, not\ b \qquad a \leftarrow b$$

We obtain the following B-SE-models of P:[2]

$$\langle \emptyset, \emptyset\rangle \quad \langle \emptyset, ac\rangle \quad \langle ac, ac\rangle$$
$$\langle \emptyset, abc\rangle \quad \langle a, abc\rangle \quad \langle abc, abc\rangle$$

Yet, the SE-models of P do not contain $\langle a, abc\rangle$, but rather $\langle c, abc\rangle$ and $\langle ac, abc\rangle$, where the latter is the SE-model corresponding to the B-SE-model $\langle a, abc\rangle$ by (ii) of Lemma 1.

Unlike SE-models, B-SE-models are not closed under program composition, i.e., $SE^B(P \cup Q) = SE^B(P) \cap SE^B(Q)$ does in general not hold. It is only true if $B = \mathcal{U}$.

Example 2. *Consider program $P = \{d \leftarrow a, c\,;\, b\}$ and $B = \{c\}$. We have $SE^B(P) = \{\langle b, b\rangle, \langle \emptyset, bc\rangle, \langle bc, bc\rangle\}$. Yet, $SE^B(\{b\}) = \{\langle b, b\rangle\}$, while $SE^B(\{d \leftarrow a, c\}) = \{\langle \emptyset, \emptyset\rangle, \langle \emptyset, c\rangle, \langle c, c\rangle\}$. Clearly, closure does not hold.*

As argued (Eiter, Fink, and Woltran 2007), this is not really surprising as otherwise, for $B = \emptyset$, answer set semantics would be monotonic. Still, we can show that B-SE-models permit determining answer sets under composition.

Proposition 2. *Let P be a program and B and F sets of atoms s.t. $F \subseteq B$. We have $Y \in AS(P \cup F)$ iff i) $F \subseteq Y$; ii) $\langle Y, Y\rangle \in SE^B(P)$; and iii) for each M with $F \subseteq M \subset Y$, $\langle M, Y\rangle \notin SE^B(P)$.*

[1] For full versions of proofs throughout the paper see https://www.dbai.tuwien.ac.at/user/saribat/pub/kr26-supp.pdf.

[2] We follow a usual convention and abbreviate sets in (B-)SE-interpretations, such as $\{a, c\}$ with the sequence of its elements.

Note that for $F = \emptyset$, this yields $Y \in AS(P)$ iff $\langle Y, Y\rangle \in SE^B(P)$ and $\langle X, Y\rangle \notin SE^B(P)$ for any $X \subset Y$.

The characterization of uniform equivalence of programs can be captured by their *UE-models* (Eiter, Fink, and Woltran 2007) which are defined as $UE(P) = \{\langle Y, Y\rangle \in SE(P)\} \cup max_{\geq}\{\langle X, Y\rangle \in SE(P)$ and $X \subset Y\}$ where $max_{\geq}$ chooses the maximal elements of a set according to a given relation $\geq$, and $\langle X', Y'\rangle \geq \langle X, Y\rangle$ iff $Y' = Y$ and $X \subseteq X'$. Then $P_1 \equiv_u P_2$ iff $UE(P_1) = UE(P_2)$. Similar to relativized B-SE models, relativized B-UE-models, which are defined analogous to UE-models (Eiter, Fink, and Woltran 2007), allow to consider equivalence under the addition of facts chosen from a set B. DLPs P_1, P_2 are uniformly equivalent relative to B iff $UE^B(P_1) = UE^B(P_2)$.

3 Relativized Uniform Abstractions

Uniform abstractions (Saribatur et al. 2024) admit generalizing atoms in a program to common terms while preserving the answer sets of the original program in the resulting abstraction, independently of the set of facts that is added. This aligns with uniform equivalence and the idea of maintaining an ASP encoding and only changing the instance data. It turns out that this notion sometimes may be too strong.

Example 3. *Consider the course plan of a computer science master's course, where a number of different courses are offered in different terms, in part with overlapping schedules, and students can choose their courses, while abiding with a number of requirements. Suppose one time slot offers doing the KRR course (a) or, alternatively, the software systems seminar (b). Also, doing the KRR course ensures doing an AI course (c). We can represent this in a program P:*

$$a \leftarrow not\ b \qquad b \leftarrow not\ a \qquad c \leftarrow a$$

Now, suppose we want to generalize this small program, focusing on the aspect of doing core courses from certain areas and that an AI course is such a core course, i.e., we want to cluster a and c into the concept core course (k). Such an abstracted program over b and k would be required to have the original answer sets $\{a, c\}$ and $\{b\}$, projected under the clustering, i.e., $\{k\}$ and $\{b\}$, but also ensure correspondence of answer sets under any addition of facts. Notably, since a and c are conjoined, adding them to P should yield the same result after clustering. However, adding a to P yields a single answer set, $\{a, c\}$, corresponding to $\{k\}$ after clustering, while adding c to P yields two answer sets, $\{a, c\}$ and $\{b, c\}$, corresponding to $\{k\}$ and $\{k, b\}$ after clustering, and therefore no abstracted program can exist. Yet, the only atoms we would want to add to such a program are the actual courses the student takes, not those that represent auxiliary concepts.

A solution to the problem in the previous example would be not admitting to add all possible atoms as facts, but rather only those from a predetermined set.

Before we introduce our corresponding new notion, we formalize mappings following (Saribatur et al. 2024).

Definition 2. *Given sets of atoms $\mathcal{U}$, $\mathcal{U}'$ with $|\mathcal{U}| \geq |\mathcal{U}'|$, a* mapping *$m$ is a surjective function $m : \mathcal{U} \mapsto \mathcal{U}'$.*

For a set S of atoms the inverse of the mapping is defined as $m^{-1}(S) = \{a \mid m(a) \in S\}$. Mappings and their inverses can also be applied to sets of sets of atoms, SE-models, sets of rules/facts, i.e., programs, in a pointwise manner.[3]

Given a mapping $m : \mathcal{U} \mapsto \mathcal{U}'$, we can consider the largest set $[Y]_m \subseteq \mathcal{U}$ that has the same image over m as Y, which includes all clustered atoms that have some representative in Y. Formally, given $Y \subseteq \mathcal{U}$, $[Y]_m = m^{-1}(m(Y))$.

We can then identify the subset of $\mathcal{U}$ of *clustered atoms* $\mathcal{U}_{cld} = \{u \in \mathcal{U} : |[\{u\}]_m| > 1\}$, and the subset of $\mathcal{U}'$ of *cluster atoms* $\mathcal{U}'_{cl} = \{m(u) : u \in \mathcal{U}_{cld}\}$. In the remainder, w.l.o.g., we only consider mappings m such that all elements in $\mathcal{U} \setminus \mathcal{U}_{cld}$ are mapped to themselves, and we represent for m only explicitly the atoms that are clustered.

Example 4. *Take $\mathcal{U} = \{a, b, c\}$ for the program P from Ex. 3 with $m : \{a, c\} \mapsto k$. Then, $\{a, b\}_{cld} = \{a\}$, $\{b, k\}_{cl} = \{k\}$, $[\{a, b\}]_m = \{a, b, c\}$, and $[\{b\}]_m = \{b\}$.*

We can now introduce relativized uniform abstractions:

Definition 3. *Given a program P (over $\mathcal{U}$), a set $B \subseteq \mathcal{U}$, and a mapping m, Q (over $\mathcal{U}'$) is a* (uniform) B-m-abstraction of P *if, for any set F of facts over B, we have*

$$m(AS(P \cup F)) = AS(Q \cup m(F)). \tag{1}$$

If such a Q exists, we say P is uniform B-m-abstractable.

It is easy to observe that this generalizes the notion of uniform abstraction (Saribatur et al. 2024).

Proposition 3. *P is uniform m-abstractable iff P is uniform $\mathcal{U}$-m-abstractable.*

It is also clear that for the trivial mapping, i.e., $\mathcal{U}_{cld} = \emptyset$, this corresponds to relativized uniform equivalence.

Corollary 4. *Given a program P over $\mathcal{U}$, a set $B \subseteq \mathcal{U}$, and a mapping m s.t. $\mathcal{U}_{cld} = \emptyset$, Q is a a uniform B-m-abstraction of P iff $P \equiv_u^B Q$.*

As we have observed in Ex. 3, abstractions may not be achievable for P if a mapping clusters atoms that, when added separately to P, yield different answer sets. The following notion captures when such problems do not arise.

Definition 4. *Given a program P, a set B of atoms, and a mapping m, we say that P is* B-m-preserving (of answer sets) *if it satisfies the following:*
For all sets $Z, Z' \subseteq B$ s.t. $m(Z') = m(Z)$:

$$m(AS(P \cup Z)) = m(AS(P \cup Z')) \tag{2}$$

Note that Definition 4 technically gives the following.

Lemma 5. *P is* B-m-preserving *iff it satisfies the following: For all Y and all sets of facts $Z, Z' \subseteq B$ s.t. $m(Z')=m(Z)$:*

$$\text{If } Y \in AS(P \cup Z), \text{ then}$$
$$\exists Y' \text{ s.t. } m(Y') = m(Y) \text{ and } Y' \in AS(P \cup Z') \tag{3}$$

In Ex. 3, P is not $\{a, c\}$-m-preserving. Yet, if P is B-m-abstractable, then it is B-m-preserving, i.e., we can establish a necessary condition.

[3]For individual atoms or facts, we may abuse notation and apply m and its inverse directly, and, unless stated otherwise, domain and codomain of m are always $\mathcal{U}$ and $\mathcal{U}'$ respectively.

Proposition 6. *If there is a B-m-abstraction of P, then, P is B-m-preserving.*

Proof. Suppose Q is a B-m-abstraction of P, and consider any Y, and any sets of facts $Z \subseteq B$ and $Z' \subseteq B$ such that $m(Z') = m(Z)$ and $Y \in AS(P \cup Z)$. The latter, by (1) (of Def. 3), implies that $m(Y) \in AS(Q \cup m(Z))$. Since $m(Z') = m(Z)$ we have that $m(Y) \in AS(Q \cup m(Z'))$. Finally, using again (1), we can conclude that there is some $Y' \in AS(P \cup Z')$ such that $m(Y') = m(Y)$. □

Observe that for a mapping m with $\mathcal{U}_{cld} \cap B = \emptyset$, any program becomes B-m preserving, as $Z = Z'$.

4 Characterization of Abstractions

In this section, we provide necessary and sufficient conditions for relativized abstractability and a semantic characterization of relativized abstractions.

These conditions generalize the ones introduced for uniform abstractions in Proposition 3 of (Saribatur et al. 2024). The former conditions, intuitively, capture the correspondence of answer sets between a program and its abstraction on the level of SE-models. Namely, any non-total SE-model $\langle X, Y\rangle$ of P being abstracted into a total one requires the existence of another total SE-model that admits an answer set if X is added to P ($\Delta^m_{u_1}$); for any total SE-model $\langle Y, Y\rangle$ of P, if adding X to P would turn Y into an answer set, then this behavior must be mirrored for any X' that coincides with X under abstraction ($\Delta^m_{u_2}$); and if a total SE-model contains any clustered atoms, then there also is a total SE-model containing all of those abstracted into them ($\Delta^m_{u_3}$).

For relativized abstractions, instead of determining such conditions on SE-models to identify those answer sets for all possible sets F of facts to be added, we are interested only in the cases when adding sets F of facts over B. Therefore, B-SE-models are more suitable to provide such conditions.

However, as we have seen in Section 2, B-SE models are not closed under program composition and there is no exact match between the SE-models and the B-SE-models of a program, which complicates such characterization. It turns out, though, that lifting the conditions from SE-models to B-SE-models works because of the restrictions imposed by adding only atoms over B, as we illustrate next for $\Delta^m_{u_1}$.

Example 5. *Consider program P from Ex. 1. If we were to cluster a and b to k, then, seemingly, there is no non-total B-SE-model that under such a mapping would be abstracted into a total one. (Recall that such abstraction of a B-SE-model can be achieved simply by applying the mapping to the model itself.) Yet, there is a non-total SE-model ($\langle ac, abc\rangle$) that can be abstracted into a total one, which raises the question if and how this has to be taken into consideration. It turns out that such condition for SE-models would only be meaningful for $(X', Y) \in SE(P)$ with $X' \subset (Y \cap B)$, as, otherwise, X' cannot be added to P anyway (by Definition 3) and no answer set will result from it. In fact, as we will show next, such lifting to B-SE models can be achieved without referring to SE-models.*

Similar observations can be obtained for the cases of $\Delta^m_{u_2}$ and $\Delta^m_{u_3}$. For the former, to facilitate such characterization,

we can establish a tight connection between SE-models and B-SE-models regarding the non-existence of non-total models.

To that end, we introduce notation to refer to sets X where a non-total (B-)SE-model of a given Y that is equal or larger than X does not exist. Formally, $\nexists mod_{\supseteq}(X,Y)$ (resp. $\nexists mod^B{}_{\supseteq}(X,Y)$) represents that, for each M with $X \subseteq M \subset Y$, $\langle M,Y\rangle \notin SE(P)$ (resp. $\langle M,Y\rangle \notin SE^B(P)$). We obtain the following tight connection.

Lemma 7. *Let P be a program, B a set of atoms, $\langle Y,Y\rangle \in SE^B(P)$, and $X \subseteq Y \cap B$. Then $\nexists mod^B{}_{\supseteq}(X,Y)$ implies $\nexists mod_{\supseteq}(X,Y)$.*

We can now state the necessary conditions for relativized abstractability based on B-SE-models.

Proposition 8. *If there is a B-m-abstraction of P, then P satisfies the following:*

$\Delta^{B,m}_{u_1}$: *For each $\langle X,Y\rangle \in SE^B(P)$ with $X \subset Y$ and $m(X) = m(Y)$, there exists $Y' \supseteq X$ with $m(Y') = m(Y)$, $\langle Y',Y'\rangle \in SE^B(P)$ and $\nexists mod^B{}_{\supseteq}(X,Y')$.*

$\Delta^{B,m}_{u_2}$: *For each X,Y,X' with $X \subset (Y \cap B)$, $\langle Y,Y\rangle \in SE^B(P)$, $\nexists mod^B{}_{\supseteq}(X,Y)$ and $m(X) = m(X')$, there exists $\langle Y',Y'\rangle \in SE^B(P)$ with $m(Y') = m(Y)$, $X' \subseteq Y'$ and $\nexists mod^B{}_{\supseteq}(X',Y')$.*

$\Delta^{B,m}_{u_3}$: *$\langle Y,Y\rangle \in SE^B(P)$ implies $\langle Y \cup C', Y \cup C'\rangle \in SE^B(P)$ for some C' s.t. $([Y_{cld}]_m \cap B) \subseteq C' \subseteq [Y_{cld}]_m$.*

Proof. Let Q be a B-m-abstraction of P.

$\Delta^{B,m}_{u_1}$: Consider $\langle X,Y\rangle \in SE^B(P)$ with $X \subset Y$ and $m(X) = m(Y)$. This, together with $X \subset (Y \cap B)$ (Def. 1), implies $Y \subseteq B$. Also, by Lemma 1, $\langle X',Y\rangle \in SE(P)$, for some $X' \subseteq Y$ with $(X' \cap B) = X$. Then $m(X) = m(X')$, and also $m(Y \cap B) = m(X' \cap B)$. We also have $\langle Y,Y\rangle \in SE^B(P)$, and thus, by Prop. 2, $Y \in AS(P \cup (Y \cap B))$. Given Q, by Lemma 5 and Prop. 6, there exists Y' s.t. $m(Y') = m(Y)$ and $Y' \in AS(P \cup (X' \cap B))$. We check the three conditions. 1. Since $X \subset (Y \cap B)$, $X \subset B$, and since $X' \cap B = X$, $X \subseteq X'$. Also, $m(X') = m(Y) = m(Y')$, so indeed $X \subseteq Y'$ holds. 2. Clearly, $\langle Y',Y'\rangle \in SE(P \cup (X' \cap B))$ and $\langle X'',Y'\rangle \notin SE(P \cup (X' \cap B))$ for any $X'' \subset Y'$. Then, $\langle Y',Y'\rangle \in SE(P)$, and, for any $X'' \subset Y'$ with $(X'' \cap B) = (X' \cap B)$, $\langle X'',Y'\rangle \notin SE(P)$. As $m(X' \cap B) = m(Y \cap B)$, by Def. 1, $\langle Y',Y'\rangle \in SE^B(P)$ as required. 3. Assume there exists M such that $X \subseteq M \subset Y'$, $\langle M,Y'\rangle \in SE^B(P)$. By (ii) of Lemma 1, there is $\langle X''',Y'\rangle \in SE(P)$, for some $X''' \subseteq Y'$ with $(X''' \cap B) = M$. We derive a contradiction to our observation in part 2. above, and thus $\nexists mod^B{}_{\supseteq}(X,Y')$.

$\Delta^{B,m}_{u_2}$: Consider X,Y,X' with $X \subset (Y \cap B)$, $\langle Y,Y\rangle \in SE^B(P)$, $\nexists mod^B{}_{\supseteq}(X,Y)$, and $m(X) = m(X')$. By (i) of Lemma 1, $\langle Y,Y\rangle \in SE(P)$. By Lemma 7, $\nexists mod_{\supseteq}(X,Y)$, and therefore $Y \in AS(P \cup X)$. By Prop. 6, and since $m(X) = m(X')$, there is Y' s.t. $m(Y') = m(Y)$ and $Y' \in AS(P \cup X')$. Then, $\langle Y',Y'\rangle \in SE(P \cup X')$ and $\langle X'',Y'\rangle \notin SE(P \cup X')$ for any $X'' \subset Y'$. Then, $\nexists mod_{\supseteq}(X',Y')$ and $\langle Y',Y'\rangle \in SE(P)$, and, for any $X'' \subset Y'$ with $(X'' \cap B) = (Y' \cap B)$, $\langle X'',Y'\rangle \notin SE(P)$. By Def. 1, $\langle Y',Y'\rangle \in SE^B(P)$ as required. Also, as $Y' \in AS(P \cup X')$, $X' \subseteq Y'$. Finally, assume there is M such that $X' \subseteq M \subset Y'$, $\langle M,Y'\rangle \in SE^B(P)$. By (ii) of Lemma 1, there is $\langle X''',Y\rangle \in SE(P)$, for some $X''' \subseteq Y$ with $(X''' \cap B) = M$. We derive a contradiction to our observation $\nexists mod_{\supseteq}(X',Y')$, and thus $\nexists mod^B{}_{\supseteq}(X',Y')$.

$\Delta^{B,m}_{u_3}$: Consider $\langle Y,Y\rangle \in SE^B(P)$. By Prop. 2, $Y \in AS(P \cup (Y \cap B))$. Given Q, by Prop. 6, for any set of facts Z' s.t. $m(Z') = m(Y \cap B)$, there exists Y' s.t. $m(Y') = m(Y)$ and $Y' \in AS(P \cup Z')$. We know that $m(Y \cup C'') = m(Y)$ and $m((Y \cup C'') \cap B) = m(Y \cap B)$ for any C'' s.t. $([Y_{cld}]_m \cap B) \subseteq C'' \subseteq [Y_{cld}]_m$. Thus, $(Y \cup C'') \in AS(P \cup ((Y \cup C'') \cap B))$. Then, $\langle Y \cup C'', Y \cup C''\rangle \in SE(P \cup ((Y \cup C'') \cap B))$ and $\langle X, Y \cup C''\rangle \notin SE(P \cup ((Y \cup C'') \cap B))$ for any $X \subset (Y \cup C'')$. We obtain that $\langle Y \cup C'', Y \cup C''\rangle \in SE(P)$ and that, for any $X \subset (Y \cup C'')$ with $(X \cap B) = ((Y \cup C'') \cap B)$, $\langle X, Y \cup C''\rangle \notin SE(P)$. By Def. 1, $\langle (Y \cup C''), (Y \cup C'')\rangle \in SE^B(P)$ for such C''. □

We use the abbreviation that P *satisfies* $\Delta^{B,m}_u$ iff it satisfies the conditions $\Delta^{B,m}_{u_i}$ for $1 \leq i \leq 3$.

Condition $\Delta^{B,m}_{u_3}$ deviates a bit from its correspondent $\Delta^{m}_{u_3}$ for uniform abstraction (Saribatur et al. 2024), which, in comparison, is quite simpler. The following example illustrates why this is the case.

Example 6. *Consider the following program P*

$$a \leftarrow not\ not\ a \qquad b \leftarrow not\ not\ b$$
$$d \leftarrow not\ not\ d \qquad c \leftarrow a$$

with answer sets $\emptyset, \{b\}, \{d\}, \{a,c\}, \{a,b,c\}, \{a,c,d\}$, and $\{a,b,c,d\}$. Take $m : \{a,c,d\} \rightarrow k$. When adding $\emptyset$ to P, the resulting program must have the answer sets $\{\}, \{b\}, \{k\}, \{b,k\}$. A suitable Q would be $\{b \leftarrow not\ not\ b; k \leftarrow not\ not\ k\}$. Now, if we consider $B = \{a,b,c,d\}$, then $\Delta^{B,m}_{u_3}$ is indeed valid. We would expect that the same holds true for any $B' \subset B$. However, with the equality condition, for the case of $B' = \{a,b,d\}$, this fails. The reason is: in this restricted setting, there is a B'-SE model $\langle d,d\rangle$ which for B' requires exactly $\langle ad,ad\rangle$ to be a B'-SE model (it is not, not even if we consider B itself). For B, the construction maintains c, i.e., it requires a model $\langle acd,acd\rangle$ which exists. The relaxed condition admits that C' be such that it necessarily contains all the clustered atoms that occur in B, but also other clustered atoms, not contained in B.

For $B = \mathcal{U}$, the conditions $\Delta^{B,m}_u$ exactly match the conditions of uniform abstraction (Saribatur et al. 2024).

On the other hand, if B is such that $\mathcal{U}_{cld} \cap B = \emptyset$, the conditions $\Delta^{B,m}_u$ become immaterial.

Proposition 9. *Let P be a program, m a mapping, and $B \subseteq \mathcal{U}$ s.t. $B \cap \mathcal{U}_{cld} = \emptyset$. Then, P satisfies $\Delta^{B,m}_u$.*

We now proceed by constructing a set of UE-models aimed to represent B-m-abstractions. UE-models are indeed adequate here as they capture uniform equivalence, i.e., equivalence under addition of facts.

The main idea of the construction is to first capture all total B-SE-models of the given program, as well as certain non-total B-SE-models of it, and then effectively applying m to these models.

To that end, we first collect all those non-total B-SE models $\langle X, Y\rangle$ that, for each Y' that is mapped to Y and for which a B-SE-model $\langle Y', Y'\rangle$ exists, have a non-total relativized UE-model that contains an X' compatible with X under m. Intuitively X is included as non-total model if no matter what variant of X under abstraction we add to P, it does not result in an answer set. As we are interested in the largest possible such cases, B-UE-models with their inherent maximality condition are well-suited here.

Definition 5. *Let P be program, B a set of atoms, $m\colon \mathcal{U}\to\mathcal{U}'$ a mapping, and $Y \subseteq \mathcal{U}'$:*

$$\begin{aligned} S_Y^{B,m}(P) = max_{\geq}\{X \mid\ & m(X) \subset (Y \cap m(B)) \text{ and} \\ & \forall Y' \subseteq \mathcal{U} \text{ s.t. } m(Y') = Y \text{ and} \\ & \langle Y', Y'\rangle \in SE^B(P) \text{ and} \\ & \forall X' \subset (Y' \cap B) \text{ s.t. } m(X') = m(X) : \\ & \exists M \text{ s.t. } X' \subseteq M \subset Y' \text{ and} \\ & \langle M, Y'\rangle \in UE^B(P)\} \end{aligned}$$

Example 7. *Recall program P from Ex. 1 and consider $m : \{a, b\} \to k$. For $Y = \emptyset$, no subset of Y can exist and $S_\emptyset^{B,m}(P) = \emptyset$. The only other option is $Y = \{k, c\}$, and we have two Y' with $m(Y') = Y$, and $S_{\{k,c\}}^{B,m}(P) = \{\emptyset\}$. Note that adding $\emptyset$ does indeed not create an answer set of P containing $\{a, c\}$, while adding $\{a\}$ would.*

With this in place, we can determine a set of models as outlined, defining B-relativized m-UE-models, as the B-relativized UE-models under mapping m.

Definition 6. *Let P be program and m a mapping. The set of B-*relativized m-UE-models of P, *denoted $UE_m^B(P)$, is defined as:*

$$\begin{aligned} &\{\langle m(X), m(Y)\rangle \mid X \in S_{m(Y)}^{B,m}(P), \langle Y, Y\rangle \in SE^B(P)\} \\ &\cup \{\langle m(Y), m(Y)\rangle \mid \langle Y, Y\rangle \in SE^B(P)\} \end{aligned}$$

Example 8. *(Ex. 7 ctd.) By construction, we obtain $UE_m^B(P) = \{\langle\emptyset, \emptyset\rangle, \langle\emptyset, kc\rangle, \langle kc, kc\rangle\}$.*

We can show that for a uniform m-abstraction of a program, its B-relativized m-UE-models and the $m(B)$-relativized UE-models coincide.

Proposition 10. *Let P be program and m a mapping. If Q is a B-m-abstraction of P, then it satisfies*

$$UE_m^B(P) = UE^{m(B)}(Q). \tag{4}$$

Proof sketch. For the inclusion $UE_m^B(P) \subseteq UE^{m(B)}(Q)$, we assume that $\langle X, Y\rangle \in UE_m^B(P)$, but $\langle X, Y\rangle \notin UE^{m(B)}(Q)$. The non-trivial case is when $X \subset Y$, and we have two cases: there exists $\langle M, Y\rangle \in UE^{m(B)}(Q)$ s.t. $X \subset M \subset Y$; or $\nexists mod_{\supseteq}^{m(B)}(X, Y)$. In both cases, we reach a contradiction with $X^* \in S_Y^{B,m}(P)$ with $m(X^*) = X$.

For the reverse inclusion, assume $\langle X, Y\rangle \in UE^{m(B)}(Q)$, but suppose $\langle X, Y\rangle \notin UE_m^B(P)$. The non-trivial case is $X \subset Y$. We consider $X^* = m^{-1}(X)$ and have two alternatives: X^* does not satisfy the conditions of $S_Y^{B,m}(P)$; or X^* is not maximal among those that satisfy $S_Y^{B,m}(P)$. In both cases this contradicts $\langle X, Y\rangle \in UE^{m(B)}(Q)$. □

With Cor. 4, for a trivial mapping m, this also yields $P_1 \equiv_u^B P_2$ iff $UE^B(P_1) = UE^B(P_2)$ for ELPs P_1, P_2.

More importantly, we obtain sufficient conditions for relativized uniform m-abstractions.

Theorem 11. *Let P be program, B a set of atoms, and m a mapping. There exists a B-m-abstraction of P iff P satisfies $\Delta_u^{B,m}$.*

For this, we show that a program Q over $\mathcal{U}'$ satisfying $UE_m^B(P) = UE^{m(B)}(Q)$ is a B-m-abstraction of P.

This also allows to conclude that if $B \cap \mathcal{U}_{cld} = \emptyset$, then we can always find a m-abstraction of P w.r.t. B.

Corollary 12. *Let $B \subseteq \mathcal{U}$ and m a mapping with $B \cap \mathcal{U}_{cld} = \emptyset$, then any program is B-m-abstractable.*

5 Closure of Abstractions

In this section, we study closure of abstractions w.r.t. subclasses of logic programs. In general, even if we start with a DLP, we may end up with an abstraction which is not a DLP.

Example 9. *Let us consider again program P from Ex. 3.*

$$a \leftarrow not\ b \qquad b \leftarrow not\ a \qquad c \leftarrow a$$

Suppose the cluster mapping is $m : \{a, b\} \mapsto k$, and take $B = \{c\}$. In this case, the answer sets of P are $\{a, c\}$ and $\{b\}$, which implies that a B-m-abstraction of P must have $\{k, c\}$ and $\{k\}$ as answer sets. Yet, the existence of non-minimal answer sets immediately clarifies that no corresponding DLP exists. Still, a B-m-abstraction of P exists (though not a DLP). Consider Q:

$$c \leftarrow not\ not\ c \qquad k \leftarrow$$

It is easy to check that Q is a B-m-abstraction of P. The same is true for $m : \{b, c\} \mapsto k$ and $B = \{c\}$. In this case, the abstracted program must have $\{k, a\}$ and $\{k\}$ as answer sets. Again, there are non-minimal answer sets, so no matching DLP can exist. Still, the following extended program Q is a B-m-abstraction of P:

$$a \leftarrow not\ not\ a \qquad k \leftarrow$$

So, in both cases, although P is a DLP (in fact an NLP), and there are B-m-abstractions of P, none of these can be a DLP. Also note these two choices of m and B are representatives of the two disjoint cases when $\mathcal{U}_{cld} \not\subseteq B$: i) $\mathcal{U}_{cld} \cap B = \emptyset$ and ii) $\mathcal{U}_{cld} \cap B \neq \emptyset$, but $\mathcal{U}_{cld} \cap (\mathcal{U} \setminus B) \neq \emptyset$.

Although closure for DLPs does not hold in general, we are able to prove that if $\mathcal{U}_{cld} \subseteq B$, such result holds. For this, we recall the semantic characterization of DLPs (Eiter et al. 2013). A set $\mathcal{S}$ of SE-interpretations is said to be *complete* if, whenever $\langle X, Y\rangle \in \mathcal{S}$, then $\langle Y, Y\rangle \in \mathcal{S}$, and if, whenever $\langle X, Y\rangle \in \mathcal{S}$ and $\langle Z, Z\rangle \in \mathcal{S}$, with $Y \subset Z$, then $\langle X, Z\rangle \in \mathcal{S}$.

Proposition 13 ((Eiter et al. 2013))**.** *A set $\mathcal{S}$ of SE-interpretations is complete if and only if there is a DLP P such that $SE(P) = \mathcal{S}$.*

In the relativized case, we can also consider similar characterizations, both for B-SE-models and B-UE-models.

Definition 7. *A set $\mathcal{S}$ of B-SE-interpretations is said to be* complete *if the following conditions hold:*

- *if $\langle X, Y\rangle \in \mathcal{S}$, then $\langle Y, Y\rangle \in \mathcal{S}$, and*
- *if $\langle X, Y\rangle \in \mathcal{S}$ and $\langle Z, Z\rangle \in \mathcal{S}$, with $Y \subset Z$, then $\langle X \cap B, Z\rangle \in \mathcal{S}$.*

For B-UE-models, one extra condition is needed.

Definition 8. *A set $\mathcal{S}$ of B-SE-interpretations is said to be* quasi-complete *if the following conditions hold:*

- *if $\langle X, Y\rangle \in \mathcal{S}$, then $\langle Y, Y\rangle \in \mathcal{S}$*
- *if $\langle X, Y\rangle \in \mathcal{S}$, then $\langle X', Y\rangle \notin \mathcal{S}$ for every $X \subset X' \subset Y$.*
- *if $\langle X, Y\rangle \in \mathcal{S}$ and $\langle Z, Z\rangle \in \mathcal{S}$, with $Y \subset Z$, then there exists $M \supseteq X \cap B$ such that $\langle M, Z\rangle \in \mathcal{S}$.*

We can now present novel results that provide necessary and sufficient conditions for a set of B-SE-interpretations to be the set of models of a DLP. These follow from the results in (Eiter, Fink, and Woltran 2007) on SE^B and UE^B-models and those in (Eiter et al. 2013) on DLPs.

Proposition 14. *Let $\mathcal{S}$ be a set of B-SE-interpretations.*

- *$\mathcal{S}$ is complete iff $\mathcal{S} = SE^B(P)$ for some DLP P;*
- *$\mathcal{S}$ is quasi-complete iff $\mathcal{S} = UE^B(P)$ for some DLP P.*

We now focus on the particular case of $\mathcal{U}_{cld} \subseteq B$, which means that all clustered atoms are part of the set of atoms that can be added as additional facts to the program, when comparing answer sets. As mentioned before, in both cases of Ex. 9, i.e., $\mathcal{U}_{cld} = \{a, b\}$ with $B = \{c\}$, and $\mathcal{U}_{cld} = \{b, c\}$ with $B = \{c\}$, this condition does not hold.

We begin with a simple result showing that in this particular case, condition $\Delta^{B,m}_{u_3}$ can be written in a simpler way.

Lemma 15. *Let P be program, m a mapping, and B such that $\mathcal{U}_{cld} \subseteq B$. Then, $\Delta^{B,m}_{u_3}$ can be written as:*
$\langle Y, Y\rangle \in SE^B(P)$ implies $\langle [Y]_m, [Y]_m\rangle \in SE^B(P)$.

Observe that this matches $\Delta^m_{u_3}$ (Saribatur et al. 2024).

We now show that, in this particular case, and provided that P is a DLP, the constructed set of B-SE-interpretation $UE^B_m(P)$ satisfies the DLP characterization conditions.

Proposition 16. *Let P be a DLP, m a mapping, and B such that $\mathcal{U}_{cld} \subseteq B$. If P satisfies $\Delta^{B,m}_u$, then $UE^B_m(P)$ is quasi-complete.*

Since $UE^B_m(P)$ is the characterization of any B-m-abstraction of P, this result entails the closure of abstraction with respect to DLPs.

Theorem 17. *Let P be a DLP, m a mapping, and B s.t. $\mathcal{U}_{cld} \subseteq B$. If P satisfies $\Delta^{B,m}_u$, then there is a DLP Q that is a B-m-abstraction of P.*

For the closure w.r.t. NLPs, we need to recall that every DLP is uniformly equivalent to some NLP (Eiter et al. 2013). This, together with the fact that we only characterize the result of abstraction up to B-UE-models, implies that such closure result also applies to NLPs.

Corollary 18. *Let P be an NLP, m a mapping, and B such that $\mathcal{U}_{cld} \subseteq B$. If P satisfies $\Delta^{B,m}_u$, then there is an NLP Q such that Q is a B-m-abstraction of P.*

This also applies to the non-relativized case with $B = \mathcal{U}$ (Saribatur et al. 2024), but was not proven before.

Corollary 19. *Let P be a uniform m-abstractable program. Then, if P is a DLP (resp. NLP), then there is a DLP (resp. NLP) Q s.t. Q is an m-abstraction of P.*

6 Computing Abstractions

In this section, we investigate the construction of B-m-abstractions. For this, we consider an intuitive syntactic operator that preserves the rules of the original program as much as possible. The operator considered in (Saribatur et al. 2024) had a similar motivation, replacing occurrences of the elements mapped to the same cluster simply by that cluster atom to obtain the *mapping of* P, i.e., $m(P)$. However, it only is applicable to a subclass of DLPs where the negative bodies of all rules do not contain atoms to cluster, referred as m-*positive*. Here, we propose an operator that does not apply any such limitation on the program P to be abstracted, thereby also lifting to ELPs.

Example 10. *Consider program P and $m : \{a, b\} \to k$.*

$$b \leftarrow not\ a \qquad a \leftarrow not\ b \qquad c \leftarrow a \qquad c \leftarrow b$$

P is uniform m-abstractable, i.e., $B = \mathcal{U}$, though it is not m-positive. In this case, program $Q = \{k; c \leftarrow k\}$ is a $\mathcal{U}$-m-abstraction of P, that can be obtained by applying m to P while omitting negated atoms if there is an atom in the head of the rule they are being clustered with.

Now consider program $P = \{b \leftarrow not\ a; a \leftarrow b\}$, which is $\{b\}$-m-abstractable. Here, we have $Q = \{k \leftarrow not\ k; k \leftarrow k\}$ as the abstraction, where clustered atoms in the negative body are not omitted, as the odd cycle among the atoms needs to be preserved.

With this in mind, we can now propose our syntactic operator. For this, for a given program, let $L_{c_1}, \ldots, L_{c_l}$ denote all disjoint sets of atoms involved in an odd cycle, if exists.

Definition 9. *Given a rule $r : H(r) \leftarrow B(r)$ and a mapping m, the $m^\star$ operator applied to r, denoted by $m^\star(r)$, yields*

$$m(H(r)) \leftarrow m(B^+(r)), m(B^-(r) \setminus L), m(B^{--}(r))$$

where $L \subseteq B^-(r) \cap \mathcal{U}_{cld}$ satisfying $\forall \ell \in L\ \exists \alpha \in H(r) : m(\alpha) = m(\ell)$ and $\{\alpha, \ell\} \not\subseteq L_{c_j}$ for all j.
By applying $m^\star$ to P, *we mean $m^\star(P) \coloneqq \bigcup_{r \in P} m^\star(r)$.*

The aim of $m^\star$ is to apply the mapping to each atom in the rule, and remove the atoms from the negative body that are being clustered together with an atom from the head of the rule, except when they are involved in an odd cycle. Observe that the operator obtains the abstractions in Example 10.

Clustering inside of B We begin with showing that for the cases of only abstracting those atoms that are in the vocabulary of the added facts, i.e., $\mathcal{U}_{cld} \subseteq B$, the operator $m^\star$ can obtain the abstraction. To reach this result, it is sufficient to look at the case for P being B-m-preserving (Defn. 4) without going into the details of SE/UE-models.

Proposition 20. *Let P be a program, B a set of atoms, and m a mapping with $\mathcal{U}_{cld} \subseteq B$. If P is B-m-preserving, then $Q = m^\star(P)$ is a B-m-abstraction of P.*

Proof (Sketch). We prove by contradiction, assuming that, for some $F \subseteq B$ of facts, we have $m(AS(P \cup F)) \neq AS(m^\star(P) \cup m(F))$. Since P is B-m-preserving, we choose $F' = [F]_m \cap B$ which contains all of those atoms clustered in $m(F)$ in relation with B. We do a case analysis for all cases that invalidate the equality, taking into account the rules in P that are modified by $m^\star$, reaching a contradiction for each case. We make use of the fact that $\mathcal{U}_{cld} \subseteq B$ when reaching a contradiction to Defn. 4 while traversing over F's (while remaining within B) that agree on the mapping, but turn out to not agree on the answer sets. □

Note that this establishes that for $\mathcal{U}_{cld} \subseteq B$, P being B-m-preserving is also a sufficient condition for obtaining a relativized abstraction.

Corollary 21. *For a mapping m with $\mathcal{U}_{cld} \subseteq B$, the following are equivalent:*

(i) P is B-m-preserving.
(ii) $m^\star(P)$ is a B-m-abstraction of P.
(iii) There is a B-m-abstraction of P.
(iv) P satisfies $\Delta_u^{m,B}$.

Clustering outside of B Even though it is always possible to cluster when clustering atoms outside of B (Corollary 12), simple syntactic clustering might not be possible. We observe that this occurs when the clustering loses the distinct behavior of the atoms.

Example 11 (Example 9 ctd.)**.** *For $B = \{c\}$, the abstraction contains double negation, thus cannot be achieved with $m^\star$. We see that $AS(P \cup \{a\}) = \{\{a, c\}\}$ while $AS(P \cup \{b\}) = \{\{b\}\}$ which can no longer be distinguished if a and b are clustered into k.*

Furthermore, for the cases when we cluster atoms both from inside and outside of B into one, it is not sufficient to check P being B-m-preserving to ensure whether the syntactic operator obtains the desired abstraction.

Example 12. *Consider program $P = \{d \leftarrow a, c; b\}$, mapping $m : \{a, b\} \to k$ and $B = \{b, c\}$. Here P is B-m-preserving. However $m^\star(P) = \{d \leftarrow k, c.\quad k.\}$ is not a B-m-abstraction of P, since $AS(m^\star(P) \cup \{c\}) = \{\{k, c, d\}\}$ while $AS(P \cup \{c\}) = \{\{b, c\}\}$. When renaming b in P with the clustered atom k, one loses the distinction of the rule that appears when the fact c is added. For such a P an abstraction would be $Q = \{k\}$.*

7 On Simplification and Forgetting

In this section, we show how the related notions of forgetting and simplification in ASP can be seen as a special case of relativized uniform abstractions. For this we introduce a notion of relativized uniform simplifications, that relaxes uniform simplification (Saribatur and Woltran 2023).

In order to capture removal of atoms from the vocabulary, we extend the co-domain of the mapping m with $\top$, and focus on the mapping $m' : \mathcal{U} \to \mathcal{U}' \cup \{\top\}$. Here mapping one or more elements to $\top$ is intended to represent the removal of the atoms. For simplicity, we omit $\top$ from the result of the mapping, which is aligned with the idea that the special element $\top$ corresponds to the logical constant $\top$, in which case, in a set of facts, it can indeed be removed. Since a mapping $m' : A \to \top$ can be fully determined by the set A of atoms to be removed, here we explicitly refer to A instead of m', and call these *A-omission mappings*. Moreover, given a set $X \subseteq \mathcal{U}$ we use the notation $X_{|\overline{A}}$ to refer to the projection onto the remaining atoms, that is, $X \setminus A$. This notation extends naturally to sets of sets and sets of facts.

7.1 Relativized Uniform Simplification

We start with adjusting Definition 3 to consider the special case of removing atoms from $\mathcal{U}$.

We can adapt the notion of a B-m-abstraction to a corresponding notion of omission as follows.

Definition 10. *Given $A, B \subseteq \mathcal{U}$ and a program P (over $\mathcal{U}$), program Q (over $\overline{A} = \mathcal{U} \setminus A$) is a* uniform B-A-simplification of P *if for any set $F \subseteq B$, we have*

$$AS(P \cup F)_{|\overline{A}} = AS(Q \cup F_{|\overline{A}}). \tag{5}$$

We say that P is uniform B-A-simplifiable if there exists a program Q such that (5) *holds.*

Observe that, for $B = \mathcal{U}$, this exactly amounts to the notion of uniform simplification (Saribatur and Woltran 2023).

Next, we can lift the result on the necessary and sufficient conditions for uniform B-m-abstractability to uniform B-A-simplifiability. For this, we reuse the notion of B-m-preserving (Definition 4) and Lemma 5, but now w.r.t. the set of atoms to be removed.

Definition 11. *Given a program P and sets $A, B \subseteq \mathcal{U}$ of atoms, we say that P is B-A* preserving (of answer sets) *if it satisfies the following:*
For all Y and all sets of facts $Z, Z' \subseteq B$ s.t. $Z'_{|\overline{A}} = Z_{|\overline{A}}$:

$$\begin{aligned} &\text{If } Y \in AS(P \cup Z), \text{ then} \\ &\exists Y' \text{ s.t. } Y'_{|\overline{A}} = Y_{|\overline{A}} \text{ and } Y' \in AS(P \cup Z') \end{aligned} \tag{6}$$

The observation regarding the relation with relativized simplifiability and answer set preserving then follows.

Proposition 22. *If there is a uniform A-simplification of P relative to A, then, P is B-A preserving.*

We can now present the Δ conditions for relativized simplification. Below, we may again say that P *satisfies* $\Delta_u^{B,A}$ iff it satisfies the conditions $\Delta_{u_i}^{B,A}$ for $1 \leq i \leq 3$.

Proposition 23. *If there is a uniform B-A-simplification of P, then P satisfies the following:*

$\Delta_{u_1}^{B,A}$*: For each $\langle X, Y\rangle \in SE^B(P)$ with $X \subset Y$ and $X_{|\overline{A}} = Y_{|\overline{A}}$, there exists $Y' \supseteq X$ with $Y'_{|\overline{A}} = Y_{|\overline{A}}$, $\langle Y', Y'\rangle \in SE^B(P)$ and $\nexists mod^B_{\supseteq}(X, Y')$.*

$\Delta_{u_2}^{B,A}$*: For each X, Y, X' with $X \subset (Y \cap B)$, $\langle Y, Y\rangle \in SE^B(P)$, $\nexists mod^B_{\supseteq}(X, Y)$ and $X_{|\overline{A}} = X'_{|\overline{A}}$, there exists $\langle Y', Y'\rangle \in SE^B(P)$ with $Y'_{|\overline{A}} = Y_{|\overline{A}}$, $X' \subseteq Y'$ and $\nexists mod^B_{\supseteq}(X', Y')$.*

$\Delta_{u_3}^{B,A}$: *$\langle Y, Y\rangle \in SE^B(P)$ implies $\langle Y \cup C', Y \cup C'\rangle \in SE^B(P)$ for some C' s.t. $((Y \cup A) \cap B) \subseteq C' \subseteq Y \cup A$.*

The main result on the necessary and sufficient conditions can be shown with a proof following that of Theorem 11, making use of the adjusted versions of Definitions 5 and 6 for the case of removing atoms, which we omit here due to space reasons.

Theorem 24. *Let P be program, and A, B sets of atoms. There exists a uniform B-A-simplification of P iff P satisfies $\Delta_u^{B,A}$.*

Similar to the case of clustering, when we remove atoms A belonging to B, we require that P satisfies $\Delta_u^{B,A}$ to be relativized simplifiable. However when we remove atoms that are not in B, $\Delta_u^{B,A}$ conditions are trivially satisfied, and P is always relativized simplifiable.

Corollary 25. *Let A,B be sets of atoms. Any program is uniform B-A-simplifiable for $B \subseteq \overline{A}$.*

This observation allows us to connect to forgetting next.

7.2 Relation to UP-forgetting

Let us recall the following property in the scope of forgetting in ASP (Gonçalves et al. 2019), where F is a class of forgetting operators and $\mathcal{C}$ a class of programs:

(UP) F satisfies *Uniform Persistence* if, for each $f \in F$, $P \in \mathcal{C}$ and $A \subseteq \mathcal{U}$, we have $AS(f(P, A) \cup R) = AS(P \cup R)_{|\overline{A}}$ for all sets of facts $R \in \mathcal{C}$ over $\overline{A}$.

Here, $f(P, A)$ denotes the result of forgetting about A from P. Uniform persistence is also considered for a particular forgetting instance $\langle P, A\rangle$, for $P \in \mathcal{C}$ and $A \subseteq \mathcal{U}$, denoted by **(UP)**$_{\langle P,A\rangle}$, respectively (Gonçalves et al. 2019).

It is easy to see that when setting $B = \overline{A}$ in Definition 10, the definitions coincide.

Proposition 26. *A forgetting operator f satisfies* **(UP)**$_{\langle P,A\rangle}$ *iff $f(P, A)$ is a uniform $\overline{A}$-A-simplification of P.*

In (Gonçalves et al. 2021), a general operator was introduced that can be iterated and that satisfies **(UP)**. This means that **(UP)**-forgetting is always possible. It is interesting to observe that this is in line with Corollary 25.

8 Complexity

We assume familiarity with basic concepts of complexity theory. For comprehensive details we refer to (Papadimitriou 2003; Arora and Barak 2009).

For checking whether some Q is a uniform B-m-abstraction of a program P, obtaining a lower bound is trivial by utilizing the observation in Cor. 4, where we have seen that for the trivial mapping m this check amounts to relativized uniform equivalence between P and Q (which is Π_2^P-complete). We next establish a non-trivial upper bound for this problem. We anticipate that matching lower bounds can be obtained but leave this for future work.

Theorem 27. *Given a mapping m, a program P which is B-m-abstractable, and a program Q, checking whether Q is a uniform B-m-abstraction of P is in Π_3^P and Π_2^P-hard.*

Proof. For given P, Q, B, and m, we check for the equality (1) defining Q to be a uniform B-m-abstraction. For the complementary problem, we guess a set F of facts checking $F \subseteq B$ and an interpretation I checking its containment in $AS(P \cup F)$ and the containment of $m(I)$ in $AS(Q \cup m(F))$, but that it does not hold for both. As answer set computation for propositional ELP is in Π_2^P, the equality check is contained in Σ_3^P. □

We present complexity results on deciding relativized abstractability, and start with two distinct cases on clustering.

Proposition 28. *Let P be a program $\mathcal{U}$, $B \subseteq \mathcal{U}$ a set of atoms, and m a mapping.*

(i) For $B \cap \mathcal{U}_{cld} = \emptyset$, deciding whether P is uniform B-m-abstractable is trivial.

(ii) For $\mathcal{U}_{cld} \subseteq B$, deciding whether P is uniform B-m-abstractable is in Π_3^P.

Proof. Item (i) is trivial Corollary 12. For item (ii), due to Corollary 21, it is sufficient to construct the program $m^\star(P)$, and then check whether it is a relativized uniform abstraction of P. Since the construction is done in polynomial time, we remain in Π_3^P. □

In general, complexity of this problem may increase.

Theorem 29. *Let P be a program over $\mathcal{U}$, m a mapping, and $B \subseteq \mathcal{U}$ a set of atoms. Deciding whether P is uniform B-m-abstractable is in Π_4^P and Π_3^P-hard.*

The increase in complexity, when compared to uniform abstraction, is due to relativized SE-model checking, which is D^P-complete, and thus one level higher than SE-model checking. It remains an open question whether we can confirm the gap compared to the problem of deciding abstractability for a given program P (Thm. 27) as it was not present for uniform abstractions (Saribatur et al. 2024). Note also that uniform equivalence typically incurs additional complexity compared to strong equivalence, and similarly for relativized strong equivalence (see (Eiter, Fink, and Woltran 2007)). The fact that we consider both uniform and relativized versions of equivalence, together with an abstraction mapping, accounts for the increase in complexity. In any case, if we consider P without any disjunction (but possibly double negation), then we obtain Π_3^P-completeness. This holds essentially because relativized SE-model checking is P-complete for NLPs and because, without the minimality check required for disjunctions in DLPs, relativized SE-model checking for ELPs without disjunction is the same.

9 Conclusions

We have introduced the notion of relativized uniform abstraction that is more flexible than uniform abstractions by allowing abstractions to be tailored to specific input signatures. We have lifted the logical framework to extended logic programs for wider practical applicability, while also proposing a syntactic operator to obtain an abstraction that resembles the original program as much as possible for the

case of clustering within the added facts and preserving closure within the program class. This framework also unifies existing techniques, such as uniform simplification and UP-forgetting, into a single theoretical lens.

Next steps involve lifting the UP-forgetting operator into the clustering framework, in order to have some handle on obtaining syntactic abstractions even when we cluster outside of the added facts. Our observations hint that incorporating symmetry detection might be a promising direction for identifying atoms that behave identically and automating the discovery of abstractions. Additionally, resolving the structural reasons for the complexity gap observed between checking abstractions and checking abstractability will be essential to better understand the computational limits of relativized abstractions.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. Z. G. Saribatur has been supported by the Austrian Science Fund (FWF) grant 10.55776/T1315 and the Vienna Science and Technology Fund (WWTF) grant 10.47379/ICT25044. R. Gonçalves and M. Knorr have been supported by project BIO-REVISE (2023.13327.PEX), project FAIR (2024.16987.PEX), and NOVA LINCS (UIDB/04516/2025) with the financial support of FCT.IP.

AI Declaration

The authors have not employed any Generative AI tools.

References

Arora, S., and Barak, B. 2009. *Computational complexity: a modern approach*. Cambridge University Press.

Banihashemi, B.; De Giacomo, G.; and Lespérance, Y. 2017. Abstraction in situation calculus action theories. In *Proc. of AAAI*, 1048–1055.

Baral, C., and Zhang, Y. 2005. Knowledge updates: Semantics and complexity issues. *Artificial Intelligence* 164(1-2):209–243.

Bonet, B.; Fuentetaja, R.; E-Martín, Y.; and Borrajo, D. 2019. Guarantees for sound abstractions for generalized planning. In Kraus, S., ed., *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 1566–1573. ijcai.org.

Bonet, B.; Francès, G.; and Geffner, H. 2019. Learning features and abstract actions for computing generalized plans. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 2703–2710. AAAI Press.

Cerna, D. M., and Cropper, A. 2024. Generalisation through negation and predicate invention. In *Proc. AAAI*, volume 38, 10467–10475.

Cropper, A.; Dumančić, S.; Evans, R.; and Muggleton, S. H. 2022. Inductive logic programming at 30. *ML* 111(1):147–172.

Cui, Z.; Liu, Y.; and Luo, K. 2021. A uniform abstraction framework for generalized planning. In Zhou, Z., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, 1837–1844. ijcai.org.

Eiter, T., and Fink, M. 2003. Uniform equivalence of logic programs under the stable model semantics. In *International Conference on Logic Programming*, 224–238. Springer.

Eiter, T., and Kern-Isberner, G. 2018. A brief survey on forgetting from a knowledge representation and reasoning perspective. *KI – Künstliche Intelligenz*. Online http://link.springer.com/article/10.1007/s13218-018-0564-6.

Eiter, T., and Wang, K. 2008. Semantic forgetting in answer set programming. *Artificial Intelligence* 172(14):1644–1672.

Eiter, T.; Fink, M.; Pührer, J.; Tompits, H.; and Woltran, S. 2013. Model-based recasting in answer-set programming. *Journal of Applied Non-Classical Logics* 23(1-2):75–104.

Eiter, T.; Fink, M.; and Woltran, S. 2007. Semantical characterizations and complexity of equivalences in answer set programming. *ACM Transactions on Computational Logic (TOCL)* 8(3):17.

Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proceedings of the 5th International Conference and Symposium on Logic Programming (ICLP 1988)*, 1070–1080. MIT Press.

Giacomo, G. D.; Lespérance, Y.; and Mancanelli, M. 2025. Situation calculus temporally lifted abstractions for generalized planning. In *Proc. AAAI*, 14848–14857.

Giunchiglia, F., and Walsh, T. 1992. A theory of abstraction. *AIJ* 57(2-3):323–389.

Gonçalves, R.; Janhunen, T.; Knorr, M.; Leite, J.; and Woltran, S. 2019. Forgetting in modular answer set programming. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, volume 33, 2843–2850.

Gonçalves, R.; Janhunen, T.; Knorr, M.; and Leite, J. 2021. On syntactic forgetting under uniform equivalence. In Faber, W.; Friedrich, G.; Gebser, M.; and Morak, M., eds., *Logics in Artificial Intelligence - 17th European Conference, JELIA 2021, Virtual Event, May 17-20, 2021, Proceedings*, volume 12678 of *Lecture Notes in Computer Science*, 297–312. Springer.

Illanes, L., and McIlraith, S. A. 2019. Generalized planning via abstraction: Arbitrary numbers of objects. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 7610–7618.

Janhunen, T.; Oikarinen, E.; Tompits, H.; and Woltran, S. 2009. Modularity aspects of disjunctive stable models. *J. Artif. Intell. Res.* 35:813–857.

Lang, J.; Liberatore, P.; and Marquis, P. 2003. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res.* 18:391–443.

Lomuscio, A., and Michaliszyn, J. 2014. An abstraction

technique for the verification of multi-agent systems against atl specifications. In *KR*.

Nayak, P. P., and Levy, A. Y. 1995. A semantic theory of abstractions. In *IJCAI*, 196–203. Citeseer.

Papadimitriou, C. H. 2003. *Computational complexity*. John Wiley and Sons Ltd.

Pearce, D. 1996. A new logical characterisation of stable models and answer sets. In Dix, J.; Pereira, L. M.; and Przymusinski, T. C., eds., *Non-Monotonic Extensions of Logic Programming, NMELP '96, Bad Honnef, Germany, September 5-6, 1996, Selected Papers*, volume 1216 of *Lecture Notes in Computer Science*, 57–70. Springer.

Saribatur, Z. G., and Woltran, S. 2023. Foundations for projecting away the irrelevant in asp programs. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*, 614–624. IJCAI Organization.

Saribatur, Z. G.; Knorr, M.; Gonçalves, R.; and Leite, J. 2024. On abstracting over the irrelevant in answer set programming. In Marquis, P.; Ortiz, M.; and Pagnucco, M., eds., *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*.

Turner, H. 2001. Strong equivalence for logic programs and default theories (made easy). In *Logic Programming and Nonmotonic Reasoning: 6th International Conference, LPNMR 2001 Vienna, Austria, September 17–19, 2001 Proceedings 6*, 81–92. Springer.

Visser, A. 1996. Uniform interpolation and layered bisimulation. In *Gödel'96*, volume 6 of *Lecture Notes in Logic*, 139–164. Springer Verlag.

Woltran, S. 2004. Characterizations for relativized notions of equivalence in answer set programming. In Alferes, J. J., and Leite, J. A., eds., *Logics in Artificial Intelligence, 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004, Proceedings*, volume 3229 of *Lecture Notes in Computer Science*, 161–173. Springer.